METHOD AND APPARATUS PROVIDING REMOTE REPROGRAMMING OF PROGRAMMABLE LOGIC DEVICES USING EMBEDDED JTAG PHYSICAL LAYER AND PROTOCOL

CROSS REFERENCE

This application claims benefit of United States Provisional Application No. 60/170,199, filed December 10, 1999, which is hereby incorporated by reference in its entirety.

BACKGROUND OF THE DISCLOSURE

Programmable logic devices (PLDs), such field programmable gate arrays (FPGAs) and the like, are well known. A programmable logic device comprises a complex logical element that may be programmed to effect combinational logic, sequential logic, or combined combination and sequential logic functions. Thus, a PLD allows for software modifications of sequential and/or combinational logic in a physical layer. In this manner, bug fixes, feature enhancements, and other improvements to systems incorporating PLDs may be provided via software update.

PLDs are typically updated by sending a service technician to the physical location of the PLD. The service technician will replace a programmable read-only memory (PROM) on a board or circuit including the PLD to be reprogrammed. Alternatively, an electrically programmable read-only memory (EPROM) may be reprogrammed in the standard manner by downloading new firmware from a personal computer (PC) and using a special program adapter supplied by the manufacturer of the PLD. In either case, the board, circuit, or system incorporating the PLD to be re-programmed is taken out of operation and serviced by an on-site technician.

Therefore, it is deemed to be desirable to provide a method and apparatus for remotely programming a PLD or FPGA such that on-site visits by a service technician are not required.

SUMMARY OF THE INVENTION

The disadvantages associated with the prior art are overcome by the present invention of an apparatus and method for programming a remote programmable logic device. In a first embodiment, the apparatus comprises a processing system having a first file to second file conversion program stored therein; the processing system receiving the first file from a first communications medium and transmitting the converted second file through a second communications medium in a format native to the remote programmable logic device.

Furthermore, a method corresponding to the first embodiment comprises receiving, via a first communications medium, a first file including programmable logic instructions in a non-native format, and converting the non-native format programmable logic instructions into programmable logic instructions having a format native to the remote programmable logic device. Thereafter, transmitting to the remote programmable logic device, via a second communications medium, a second file including the native format programmable logic instructions.

In a second embodiment, the apparatus for programming at least one programmable logic devices comprises at least one circuit board respectively comprising the at least one programmable logic device coupled to at least one switching circuit. A processor system is coupled to the at least one switching circuit via a board select bus and a JTAG bus, wherein the processor system executes a file in a format native to the at least one programmable logic device. The

~~native format file enables~~ the at least one switching circuit via the board select bus, ~~and then~~ programs the at least one programmable logic device via the JTAG ~~bus~~.

Furthermore, a method corresponding to the second embodiment comprises receiving at a head-end controller via a first communications medium, a file having a format native to the at least one programmable logic device. The native format file is sent to a processor system via a second communications medium. The processor system executes the native format file to identify and selectively access the at least one programmable logic devices via the board select bus. The selectively accessed programmable logic devices are then programmed via the JTAG bus.

BRIEF DESCRIPTION OF DRAWINGS

The teachings of the present invention can be readily understood by considering the following detailed description in conjunction with the accompanying drawings, in which:

FIG. 1 depicts first embodiment of a high-level block diagram of an information processing system including apparatus according to the invention;

FIG. 2 depicts a flow diagram of a first method for programming a remote programmable logic device according to the invention of FIG. 1;

FIG. 3 depicts second embodiment of a high-level block diagram of the information processing system including apparatus according to the invention; and

FIGS. 4A and 4B together depict a relational flow diagram between various functional elements of the information processing system of FIG. 3.

DESCRIPTION OF THE EMBODIMENT

FIG. 1 depicts a high-level block diagram of an information processing system including apparatus according to the invention. Specifically, FIG. 1 depicts an information processing system 100 comprising a remote source of programming data 110, a processing system 120 such as a loop server (or video switch), and a remote programmable logic device 142 such as a a gate array, field programmable gate array, field programmable logic array, read only memory, programmed array logic, programmable logic array, complex programmable logic devices, and the like. The remote source of programming data 110 is illustratively a remotely located computer workstation, which communicates with the loop server or video switch 120 via the LAN/WAN connection 112.

The processor system 120 includes a processor board 122 and an Ethernet port 138. The Ethernet port 138 is in communication with the remote source of programming data 110, such as a computer workstation or personal computer, via a network 112, such as local area network (LAN) or wide area network (WAN) denoted as LAN/WAN 112, and illustratively an Ethernet link. Furthermore, the processor board 122 comprises various components coupled to a bus 132 including at least one processor (e.g., a CPU) 124, a buffer 126, memory (e.g., RAM) 128, and support circuits 130. The memory 128 stores software for performing a file/JTAG conversion method 200.

One or more programmable logic devices (PLDs) such as field programmable gate arrays (FPGA's) 142_1 (only one PLD shown) are located within one or more functional elements 140_1 through 140_n (collectively function elements 140), such as one or more circuit boards of a system. A person skilled in the art will understand that other programmable logic devices such as a programmable read only memory (PROM), programmable logic array (PAL), and the like may

also be utilized. In one embodiment, the processor board 122 is connected to a PLD host functional element 140_1 via an input/output (I/O) port 134 such as a back plane or inter-board input/output (I/O) bus.

In an exemplary embodiment where the back plane 134 is utilized, the back plane 134 includes a plurality of signal paths 136. The signal paths 136 conform to the physical layer and other protocols provided in IEEE Standard 1149.1-1990, entitled "IEEE Standard Test Access Port and Boundary Scan Architecture," published May 21, 1990 by the Institute of Electrical & Electronic Engineers Inc. In this manner, the back plane or inter-board I/O bus provides the physical layer and protocol-compliant Joint Test Action Group (JTAG) interface.

Moreover, by utilizing the JTAG physical layer directly within the back plane of a multi-board or multi-module system, the number of pins within the back plane connections may be reduced. That is, rather than having discrete pins or electrical connections made to each of a plurality of boards, modules or sub-systems including field programmable gate arrays, a single set of contacts or signal paths adhering to the physical layer and related protocols described in IEEE Standard 1149.1 may be used to effect reprogramming operations on each of the plurality of boards, modules or sub-systems included within the system.

FIG. 2 depicts a flow diagram of a method 200 for programming one or more remote programmable logic devices 140 according to the invention. The method 200 begins in step 202 and proceeds to step 204 where a programmer or technician accesses the remote programming workstation 110 and inputs program code in a first file format native to the programming workstation 110. For example, the program code may be written in a program language such as microcode from ALTERA®, or the like. The program code may also be

generated automatically according to a rules-based epert system in response to configuration changes or othe rsystem changes, upgrades, and the like.

In step 206, the program code is then sent, via the first communications medium (e.g., Ethernet network) 112, to the processor system 120, and in step 208, the first file is received by the processor board 122 where the first file is stored in the buffer 126. As such, the processor board 122 receives a data file in a predefined file format and stores the data file in the buffer 126.

In step 210, the CPU 124 on the processor board 122 invokes a file-to-JTAG conversion program 150, which converts the programming data from the first file format to a second file in a format native to the remote programmable logic device 142 within the host functional element 140. In particular, the CPU 124 retrieves the buffered first file and executes the file conversion software program 150. That is, the file-to-JTAG conversion program 150 converts the buffered data from the originally sent storage/transmission format to the JTAG programming format. For example, a field programmable gate array manufactured, illustratively by ALTERA Corporation of San Jose, Ca., provides programming code stored in a .POF file. The POF-to-JTAG conversion program 150 converts the buffered POF file from the originally sent storage/transmission format to the JTAG programming format. Thereafter, in step 212, the CPU stores the converted data in the buffer 126.

In step 214, a command is sent by the remote source of programming data 110 to the processor board 122. The command instructs the processor board 122 to send the converted JTAG format information to the PLD 142 within the host functional element 140. Specifically, the processor board 122 transfers the converted programming data from the buffer 126 as a second file, through the physical layer

JTAG back plane 134 according to the protocol of the IEEE 1149.1 Standards document, and to the remotely located PLD 142.

In step 216, the PLD programming data that is transferred via the JTAG back plane connection 124 to the PLD, while the processor board 122 and PLD host functional element 140 are operational. In this manner, there is no need to remove the processor board 122, the loop server/video switch 120, the PLD 142, or the PLD host element 140 from service during the programming process. As such, the PLD 142 has been remotely updated with new programming information from the remote programming workstation 110.

In step 218, the PLD host functional element 140 being updated (i.e., board) is reinitiallized. The updated PLD 142 may be reinitialized via numerous techniques. In one technique, the host functional element 140 and corresponding PLD 142 cycled on and off. Powering down the board 142 being updated resets the PLD 142 so that the new configuration data within the PLD 142 can take effect after the PLD 142 is powered up again. Alternately, the host functional element 140 provides an initialization signal to the PGA via the JTAG bus such that the PGA responsibly adopts its logic configuration to the new code. Another method allows the processor 124 to provide an initialization signal to the PLD 142 via a separate communications path (not shown).

In step 218, the host functional element 140 and corresponding PLD 142 is reinitialized (e.g. power cycled) whereupon the PLD 142 begins performing the newly programmed function or functions.

The amount of time required for switching from on line to off line, and back on line is typically inconsequential to normal operation. Furthermore, the data may be buffered

prior to processing by the PLD 142 while the PLD board 140 is quickly reset, whereupon the buffered data may be supplied to the PLD 142 for processing according to the new combination and/or sequential logic functions. In step 220, the method 200 ends.

FIG. 3 depicts a high-level block diagram of an information processing system including apparatus according to an embodiment of the invention. Specifically, FIG. 3 depicts an interactive information distribution system 300 such as a video information distribution system. Such a system is described in more detail in U.S. patent application Serial No. 09/322,814, entitled "System For Interactively Distributing Information Services" and filed October 14, 1999, which is incorporated herein by reference in its entirety as fully reproduced herein.

The information distribution system 300 of FIG. 3 comprises a remote source of programming data 310 coupled to a server (or video switch) 330 via a head-end controller (HEC) 320. A video switch suitable for use in the present invention is described in detail in U.S. patent application Serial No. 09/540,178, entitled "Method And Apparatus Of Load Sharing And Improving Fault Tolerance In An Interactive Video Distribution System", filed March 31, 2000, which is hereby incorporated herein by reference in its entirety as fully reproduced herein.

The remote programming source 310 is coupled to the HEC 320 via a first communications path 319, such as a telecommunications medium (e.g., LAN/WAN connection, Internet, or the like), or by a physical medium (e.g., CD-ROM 322, which is drawn in phantom). The head-end controller is coupled to the server 330 via a second communications path such as an Ethernet connection.

The remote programming source 310 comprises various components coupled to a bus 311 including at least one

processor (e.g., a CPU) 314, memory (i.e., RAM) 316, and support circuits (e.g., I/O controllers, power supplies, and the like) 318. The memory 128 stores software 312 for performing a programmer object file (POF) to JAM byte code (JBC) conversion. The POF is a binary file created by a compiler (not shown). Furthermore, the JBC is a file containing a sequence of commands, which instructs a programmable device to perform a routine or procedure in a specific manner.

The head end controller 320 comprises one or more processors (not shown) for communicating information (e.g., video information, data, audio information, and the like) to a plurality of remote locations, such as a plurality of subscribers for video services. The head-end controller 320 is capable of receiving files such as the JBC files from the remote programming source 310. Specifically, the remote programming source 110 communicates with the head-end controller 320 via the first communications path 319.

The server/switch 330 comprises at least one processor 332 coupled to one or more circuit boards 334_1 through 334_n (collectively circuit boards 334) each having a programmable logic device 338_1 through 338_n (collectively programmable logic devices 338) such as a field programmable gate array (FPGA). A person skilled in the art will recognize that the circuit boards 334 may differ in functionality, configuration, manufacturer and the like from one another. Furthermore, different programmable logic devices may be utilized amongst the various circuit boards 334, such as gate arrays, field programmable logic arrays, read only memory, programmed array logic, programmable logic array, complex programmable logic devices, and the like.

In one embodiment each circuit board 334 is physically coupled to the at least one processor 332 via a backplane

333. The backplane 333 may be a compact PCI backplane having n slots (not shown) for receiving the one or more circuit boards 334. A backplane suitable for use in the present invention is described in detail in U.S. patent application Serial No. 09/363,670, entitled "Tightly-Coupled Disk-To-CPU Storage Server", filed July 29, 1999, which is hereby incorporated by reference herein as fully reproduced herein.

The processor 332 is electrically coupled to each circuit board 334 via board select bus 340 and a JTAG bus 342. In one embodiment, the board select bus 340 is a parallel bus, while the JTAG bus 342 is a serial bus.

Each circuit board 334 comprises circuitry including a PLD 338 and a corresponding switching circuit 336. The PLD 338 interacts with other circuitry (not shown) on the circuit card 334 to perform some function, task, or plurality of tasks. The board select bus 340 and JTAG bus 342 are coupled to inputs of the switching circuit 336 and provide input signals to the switching circuit 336. The switching circuit 336 comprises a plurality of discrete elements and/or logic gates that function as a switch to allow signals to pass from the JTAG bus through the switching circuit 336 to the PLD.

FIGS. 4A and 4B together depict a relational flow diagram 400 between various functional elements of the information processing system 300 of FIG. 3. Generally, the head-end controller receives via a first communications medium, a file having a format native to at least one programmable logic device. The head-end controller sends the native format file to a processor system (e.g., server or switch) via a second communications medium. The processor system executes the native format file to identify and selectively access the at least one programmable logic device via a first bus, and then

programs the selectively accessed programmable logic devices via a second bus.

In particular, the flow chart depicts a sequence of events, which are utilized to program a programmable logic device (PDL) such as the ROM in a field programmable gate array. The method 400 begins in step 402 where a programmer generates an programmer object file (POF) on a remotely located workstation or personal computer. The POF is a compiled binary file. In step 404, a software conversion program (e.g., JAM file conversion software) having the capability to convert the POF into a file containing JAM byte code (JBC) is initiated by the programmer at the remote computer. The JAM conversion software is an executable program specific for a particular type of programmable logic device, which builds target files from source files (e.g., POF) by utilizing specific dependency and build specification rules. Specifically, the JAM software identifies target files, examines a file system to determine those targets requiring an update, and issues OS commands to update such target files.

In step 406, the programmer transfers the JBC file to the head-end controller via a first communications path. The first communications path may be a telecommunications medium such as such as the Internet or any LAN/WAN networking channel. Alternately, the JBC file may be stored in a physical medium such as tape, CD-ROM, and the like.

In step 408, the head-end controller receives the JBC file and in step 410 the head-end controller transfers the JBC file to the server (or switch) for execution via a second communications path, (e.g., Ethernet). In step 412, the server receives the JBC file and the method 400 proceeds to step 414.

In step 414, the processor (e.g., a plurality of processors arranged in a parallel) executes the JBC file to

run the JBC program. In step 416, the JBC program identifies target files on each of the circuit boards having the programmable logic device (PDL). In particular, the JBC program is transferred over the board select bus, which is a parallel bus between the processor, and each circuit board coupled to the backplane. The executed program initially identifies those programming logic devices on the circuit boards that correspond to the JBC program (e.g., a particular programmable logic device type or family of devices of a specific manufacturer).

In step 418 the JBC program marks the identified target files (PDL's) by enabling the switching circuit on the circuit board. Specifically, the JBC program marks the PLD target files that require an upgrade (i.e., reprogramming). Once the switching circuit corresponding to the identified target files are set in an enabling mode, then, in step 420, the JBC program is transferred to the corresponding PLD's requiring an update via the JTAG bus. Specifically, in step 422, the switching circuits that have been enabled, transfer the JBC program via the JTAG bus to the PLD.

In step 424, the particular PLD's marked for an update receive the JBC program, whereupon the program is executed to modify the operational parameters according to the programming initially created by the programmer in the programmer object file (POF). After the PDL has been updated with the programming code from the JBC program, the method proceeds to step 426.

In step 426, the PLD's that were updated must be reinitialized. In particular, the server and corresponding circuit boards having the updated PLD's are rebooted to reinitialize the particular program code of the PLD's, and in step 428, the method 400 ends.

The present invention provides several advantages over the prior art. First, the invention reduces system costs in general. For example, reducing the cost of interfacing with the CPU. That is, the CPU 124 of each board 122 or module including a PLD (i.e., a CPU within PLD host functional element 140 or circuit card 334) may have input/output ports 134 configured in a common way to provide JTAG programming support. Moreover, the actual CPU 124 utilized to provide the programming may illustratively be a low cost CPU or microprocessor system.

In addition, the subject invention allows utilization of existing software to provide remote programming of the PLD devices. That is, the standard file formats used to store and transport PLD programming files is used by the remote source of programming data 110 and the LAN/WAN 112 to couple such programming files to the processor board 122 via the Ethernet port 138. Similarly, the standard JTAG physical layer and associated protocols defined in the IEEE Standard 1149.1 document are utilized to provide local transfer of re-programming information from the processor board 122 to the PLD 142 on the PLD host functional element 140.

Conversion between the respective programming and transport file format and the JTAG programming format is accomplished by a file-to-JTAG conversion program 150. As such, the programmable updates for the PLD devices 142 may be implemented without requiring a technician to manually update the PLD on-site.

Importantly, by utilizing the aforementioned JTAG physical layer and protocol to communicate between the processor board 122 and one or more PLDs 142 within the system 100, it is not necessary to include intelligence, such as a microprocessor, on the PLD host functional elements 140 within the system. In this manner, the

decreased use of microprocessors within the system allows for a reduction in system cost.

Finally, it is noted that typical back plane specifications are primarily address controlled and data movement is between boards connected to the back plane. By contrast, a back plane constructed according to the present invention includes JTAG communication signal paths such that inter-board or inter-module programming of PLDs 142 within host functional elements 140 within the system 100 is provided. By communicating with a central processing element, such as processor board 122 within the loop server or video switch 120, each of the PLDs 142 within a system 100, including the present invention, may be updated via a common JTAG bus. Moreover, by providing an Ethernet port or other communications port 138, the processor board 122 is able to receive, from a remote source of programming data 110, updated programming instructions for the PLDs to be reprogrammed.

Furthermore, one skilled in the art will recognize that these advantages are also applicable to the second embodiment and method as disclosed in FIGS. 3 and 4. Although the embodiments that incorporate the teachings of the present invention has been shown and described in detail herein, those skilled in the art can readily devise many other varied embodiments that still incorporate these teachings.